Network Working Group
Internet Draft
<draft-ietf-calsch-irip-02.txt
Expires six months from:

Andre Coutemanche/CS&T
Steve Mansour/Netscape
Pete O'Leary/Amplitude
November 19, 1998

ICalendar Real-time Interoperability Protocol (IRIP)

Status of this Memo

Abstract

This document specifies a binding from the iCalendar Transport-independent Interoperability Protocol [ITIP] to a real-time transport. Calendaring entries defined by the iCalendar Object Model [ICAL] are composed using constructs from [RFC-2045], [RFC-2046], [RFC-2047], [RFC-2048] and [RFC-2049].

This document is based on the calendaring and scheduling model defined by [ICMS].

This document is based on discussions within the Internet Engineering Task Force (IETF) Calendaring and Scheduling (CALSCH) working group. More information about the IETF CALSCH working group activities can be found on the IMC website at http://www.imc.org, the IETF website at http://www.ietf.org/html.charters/calsch-charter.html. Refer to the references within this document for further information on how to access these various documents.

Distribution of this document is unlimited. Comments and suggestions for improvement should be sent to the authors.

1 Introduction

This binding document provides the transport specific information necessary convey iCalendar Transport-independent Interoperability Protocol [ITIP] over a real-time transport.

1.1 Related Memos

Implementers will need to be familiar with several other memos that, along with this memo, form a framework for Internet calendaring and scheduling standards.

This document - specifies a real-time binding for [ITIP].

- [ICAL] specifies a core specification of objects, data types, properties and property parameters;

- [ITIP] specifies an interoperability protocol for scheduling between different implementations;

- [IMIP] specifies a messaging-based protocol binding for [ITIP].

This document does not attempt to repeat the specification of concepts or definitions from these other memos. Where possible, references are made to the memo that provides for the specification of these concepts or definitions.

1.2 Formatting Conventions

The mechanisms defined in this memo are defined in propose. In order to refer to elements of the calendaring and scheduling model, core object or interoperability protocol defined in [ICMS], [ICAL] and [ITIP] some formatting conventions have been used.

Calendaring and scheduling roles defined by [ICMS] are referred to in quoted-strings of text with the first character of each word in upper case. For example, "Organizer" refers to a role of a "Calendar User" within the scheduling protocol defined by [ITIP].

Calendar components defined by [ICAL] are referred to with capitalized, quoted-strings of text. All calendar components start with the letter "V". For example, "VEVENT" refers to the event calendar component, "VTODO" refers to the to-do calendar component and "VJOURNAL" refers to the daily journal calendar component.

Scheduling methods defined by [ITIP] are referred to with capitalized, quoted-strings of text. For example, "REQUEST" refers to the method for requesting a scheduling calendar component be created or modified, "REPLY" refers to the method a recipient of a request uses to update their status with the "Organizer" of the calendar component.

Properties defined by [ICAL] are referred to with capitalized, quoted-strings of text, followed by the word "property". For example, "ATTENDEE" property refers to the iCalendar property used to convey the calendar address of a calendar user.

Property parameters defined by [ICAL] are referred to with lower case, quoted-strings of text, followed by the word "parameter". For example,

"VALUE" parameter refers to the iCalendar property parameter used to override the default data type for a property value.

2 Architecture

[IRIP] enables real-time interoperability between scheduling systems using the iCalendar [ICAL] format for information exchange. [IRIP] is designed primarily to allow Calendar Services (CS) as defined in [ICSM] to forward real-time requests on behalf of Calendar User Agents (CUA) and receive real-time responses. The goal of [IRIP] is to allow two or more CS's to establish connections with each other. However, the design of [IRIP] does not preclude its use from CUA directly to CS. [IRIP] allows a CS to initiate a session and perform operations on behalf of multiple CUA's without the need to reauthenticate the session for each CUA.

The sections and examples below refer to a "user", a "sender", and a "receiver". For purposes of this document these terms are defined as follows:

user - the CU that initiates a request.
sender - the agent used to contact a receiving device, send commands, and receive replies.

receiver - the agent that accepts commands and sends replies.

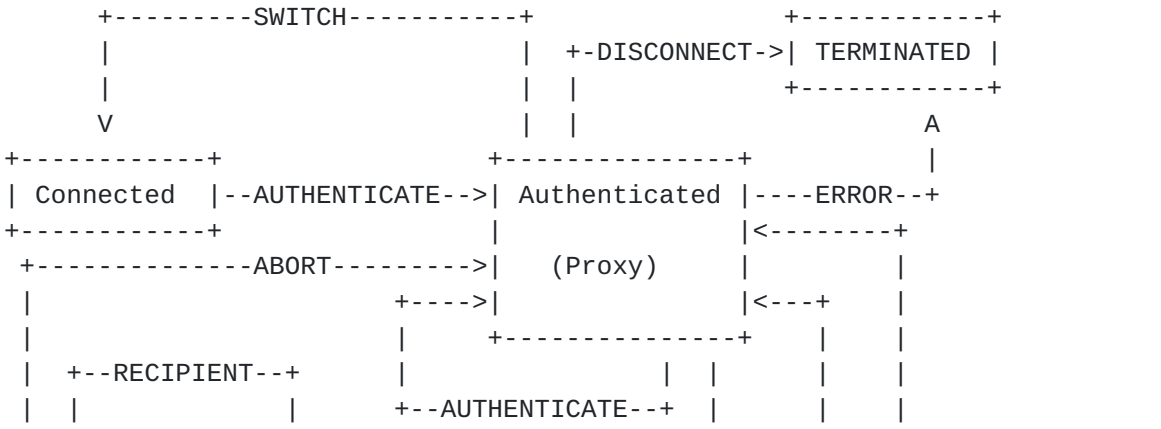
The sender and receiver can take on varying roles of CUA and CS as

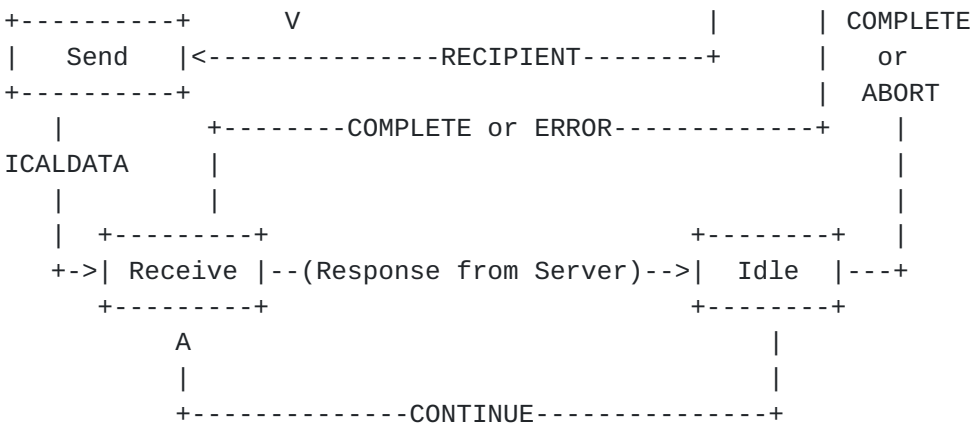
described in [ICMS].

[IRIP] allows two CS's to establish different levels of trust. When an [IRIP] connection is first established, both parties to the connection authenticate one another using the AUTHENTICATE command. The Sender can then initiate commands that the Receiver MUST interpret relative to the Sender's access control. The AUTHENTICATE command supports proxy operations via [SASL].

1.3 State Diagram

An [IRIP] session begins when a TCP/IP connection is made on port 5228. The protocol begins in the Connected state. The AUTHENTICATE command, when successful, begins the Authenticated state. From the Authenticated state, the sender can initiate a request using the RECIPIENT command. The Sender can then issues as many RECIPIENT commands as the operation in progress requires until sending an ICALDATA command. After completing the ICALDATA command, the Sender must wait for a response from the receiver. The Receiver's response can indicate that the request has been completed or that the request could not be completed in the time specified by the Sender. When the response has ended, the Sender returns to the Authenticated state where another request can be initiated. Implementations should be prepared to handle a DISCONNECT at any point in this state diagram.





1.4 Calendar Address

Calendar addresses are URIs. IRIP uses the following forms of URI.

irip://<host>:<port>/<relativeCALID>

where:

<host> is address of the computer on which the IRIP server is running

<port> is optional. Its default value is 5228.

<relativeCALID> is an identifier that uniquely identifies the calendar. There is no implied structure in a relativeCALID, it is an arbitrary string of characters. It may refer to the calendar of a user or of a resource such as a conference room.

Examples:

irip://calendar.example.com/user1
irip://calendar.example.com/conferenceRoomA
irip://calendar.example.com/89798-098-zytytasd

1.5 Bounded Latency

[IRIP] is designed so that the Sender can either obtain an immediate response from a request or discover within a known amount of time that the request cannot be completed. When the Sender initiates a command that the Receiver cannot complete within a given amount of time, the Receiver can return an error code to the Sender indicating this condition. The Sender then issues either a CONTINUE or ABORT command.

The ABORT command immediately terminates the command in progress. The

CONTINUE command instructs the Receiver to continue processing the command. The ABORT command causes the Receiver to discard the current command and return to the Authenticated state.

3 Protocol

3.1 Commands

Reply codes MAY be followed by arbitrary text. The length of the reply code, any subsequent text, and the terminating <CRLF> MUST be 1024 characters or less. The length of a RECIPIENT command and its single argument, and the terminating <CRLF> MUST be l024 characters or less. Implementations may truncate RECIPIENT lines or reply code lines that exceed 1024 characters.

In the examples below, lines preceded with "S:" refer to the Sender and lines preceded with "R:" refer to the Receiver.

3.1.1 ABORT

The ABORT command is issued by the Sender to stop an ICALDATA request from being processed further. When the latency time is specified on the ICALDATA command, the Receiver must issue a reply to the Sender within the specified time. The reply may be a reply code indicating that the server has not yet processed the request. The Sender must then tell the server whether to continue or abort.

The Sender can issue the ABORT command at any time after the ICALDATA command has been completed but before the Sender receives a reply.

Example:

```
...
S: ICALDATA:10
R: 2.0.1
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: quoted-printable
S:
S: BEGIN:VCALENDAR
S: ...
S: END:VCALENDAR
S: .
<10 seconds elapse...>
R: .
R: 2.0.2
S: ABORT
R: 2.0.3
S: <sender can now begin another command or it can disconnect>
```

The Receiver will issue the 8.2 reply code if it receives an ABORT when the ICALDATA command is not in progress. This could happen if the Sender issues an ABORT command at a point in time after the Receiver

has completed the operation and issued the reply code but before the Sender has actually received the reply code. For example:

S: ICALDATA:10
S: <an ICAL object>
S: .

<10 seconds elapse...>
S: ABORT
R: 2.0
R: 8.2
In this case, the reply code 2.0 is in response to the [ICAL] object and the reply code 8.2 is in response to the ABORT command.

3.1.2 AUTHENTICATE

The authentication mechanism used in [IRIP] is based on [SASL]. This allows the [IRIP] senders and receivers to dynamically negotiate authentication and encryption mechanisms. [SASL] defines authentication methods such as ANONYMOUS and encapsulates concepts of PROXY used in [IRIP].

The AUTHENTICATE command is used by the client to identify itself to the server. Authentication is required before the following commands can be issued:

ABORT
AUTHENTICATE
CONTINUE
DISCONNECT
ICALDATA
RECIPIENT
SWITCH

The format of the command is of the following:

AUTHENTICATE <mechanism> <initial data>

from which the standard [SASL] interchange will take place as defined in the [SASL] profile. Authentication mechanisms will differ from one server to the other, from one version to another. The CAPABILITY command must be used by the sender to determine the best authentication mechanism to use.

Example of an authentication session with kerberose version 4:

R: 2.0 Welcome IRIP Server

```
S: AUTHENTICATE KERBEROS_V4 744RTU3r#
S: sfdkjgs;lfdjg s;ldfkj gslkfdjgwrt949jsl4ns.dlngsdf
S: slkfjgsdlfjg;dslfjgdsfg
S: ;lasfgsdfg  45243 z!$14325dc
R: 2.0
```

3.1.2.1 Authentication with Proxy Access

The proxy mechanism is the ability to have data posted by an indirect source. To handle this requirement, [SASL] mechanisms have a separate "Authentication" and "authorization" identity. Thus, server A could authenticate to server B using server A's credentials with the authorization identity of user X. This effectively allows PROXY operations between servers. Some older [SASL] mechanisms do not support both authentication and authorization and therefore cannot be used when PROXY operations are required. As per the [SASL] profile, the authorization identity is the one used to determine if the operation should be allowed or not. The authentication identity ensures the transaction is originating from a trusted sender.

3.1.2.2 Authentication for Anonymous Access

SASL defines an ANONYMOUS authentication mechanism that must be used if anonymous access is to be implemented by an [IRIP] capable server. This is done by using the standard [SASL] authentication method and requesting the ANONYMOUS mechanism. The mechanism consists of a single message from the client to the server. The client sends optional trace information in the form of a human readable string. It is recommended that the trace information take one of three forms: An [RFC-822] Internet e-mail address, an opaque ASCII string which does not contain the "@" character and can be interpreted by the system administrator of the client's domain or nothing. Anonymous authentication is further described in [ANON-SASL].

The following is an example of anonymous access using an opaque ASCII string:

```
R: <listen on TCP port 5228>
S: <establish a connection to TCP port 5228>
R: 2.0
S: AUTHENTICATE ANONYMOUS
R: +
S: c21yaGM
R: 2.0
```

3.1.3 CAPABILITY

The CAPABILITY command tells the server to return a list of capabilities it supports. The server must return a CAPABILITY response with "IRIPrev1" as one of the listed capabilities. The CAPABILITY command can be issued in any connection state. The response may differ depending on the current state of the connection. The responses may also differ depending upon the authenticated user.

The format of the capabilities response is a series of lines with the form <name>[=<value>]. Each name-value pair is delimited by a <CRLF> character sequence. The sequence <CRLF>.<CRLF> followed by a reply code terminates the response.

Example:

```
S: CAPABILITY
R: CAPABILITY IRIPrev1
R: AUTH=KERBEROS_V4
R: AUTH=PLAIN
R: .
R: 2.0
```

The table below summarizes the information available response to a CAPABILITY command.

Capability	Occurs	Description
IRIPrev1	1	Revision of IRIP, must be "IRIPrev1"
AUTH	0+	Authentication mechanism(s) supported

MAXICALOBJECTSIZE	0 or 1	An integer value that specifies The largest ICAL object the server will accept. Objects larger than this will be rejected.
MAXDATE	0 or 1	The datetime value beyond which the server cannot accept.
MINDATE	0 or 1	The datetime value prior to which the server cannot accept.

3.1.4 CONTINUE

The CONTINUE command is issued by the Sender to allow an ICALDATA request to continue being processed. When the latency time is specified on the ICALDATA command, the Receiver must issue a reply to the Sender within the specified time. The reply could be a reply code indicating that the server has not yet processed the request. The Sender must then tell the server whether to continue or abort the command in progress.

The CONTINUE has the following form:

```
CONTINUE[:latencyTime]
```

If the optional latencyTime is present, it is a positive integer that specifies the maximum number of seconds the client will wait for the next response. If it is omitted, the receiver waits an indefinite period of time for the response.

In this example, the Sender requests some sort of response from the server every 10 seconds.

```
...
S: ICALDATA:10
R: 2.0.1
S: BEGIN:VCALENDAR
<etc>
S: END:VCALENDAR
S: .
<after 10 seconds...>
R: .
R: 2.0.2  Reply Pending
S: CONTINUE:10
R: BEGIN:VCALENDAR
<etc>
R: END:VCALENDAR
R: .
R: 2.0
```

3.1.5 DISCONNECT

The DISCONNECT command signals the end of communication between the Sender and Receiver. It can be sent from any state.

Example:

```
S: DISCONNECT
R: 2.0
```

3.1.6 ICALDATA

The ICALDATA command is used specify the iCalendar Object that is to be delivered to one or more recipients specified in the RECIPIENT command. The format of the command is:

```
S: ICALDATA[:latencyTime]
R: 2.0.1
S: <MIME encapsulated ITIP Message>
S: .
R: <MIME encapsulated ITIP Message>
R: .
R: <reply code>
```

The optional latencyTime value specifies the maximum number of seconds the Sender can wait for a reply. If it is not present, the client places no time limit on the server for a reply. A reply code of 2.0.1 indicates that the [ITIP] message data can be sent. When the entire message has been sent, the sender terminates sending data with the special sequence <CRLF>.<CRLF>. The receiver reply may optionally contain an ITIP message followed by the special sequence <CRLF>.<CRLF> followed by a reply code. Only the [ITIP] message is optional in the reply, the <CRLF>.<CRLF> sequence must be present.

```
S: ICALDATA
R: 2.0.1
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
<etc.>
S: END:VCALENDAR
S: .
R: .
R: 2.0
```

3.1.7 RECIPIENT

The RECIPIENT command is used to identify a recipient of the iCalendar Object. Use multiple RECIPIENT commands to specify multiple recipients. The command format is

```
RECIPIENT <calendar address>
```

A response code of 2.0 indicates that the calendar address is available for [ITIP] messages. If the receiver does not accept [ITIP] messages for the specified calendar address, it may respond with [ITIP] reply code 5.3 to indicate that the calendar address is unknown or the IRIP referral reply code, 10.1, and supply the new calendar address. In either case, the IRIP server does not deliver the [ITIP] message when

the reply code is 5.3 or 10.1.

3.1.8 SWITCH

The SWITCH command is used to allow the Sender and Receiver to change roles. Its format is:

SWITCH

The SWITCH command is useful in environments where the firewall of a Sender would not allow the Receiver to initiate a connection. The SWITCH command is issued by the Sender to give the Receiver the opportunity to take the role of the Sender. The Sender must be in the authenticated state before the SWITCH command can be used.

The Receiver must respond in one of the following fashions:

* send an OK reply and take on the role of Sender
* send a error reply indicating refusal and retain the role of Receiver

If program-A is currently the Sender and sends the SWITCH command and receives an OK reply then program-A becomes the Receiver. Program-A is then in its initial state and sends a service ready greeting message.

If program-B is currently the Receiver and sends an OK reply in response to a SWITCH command then program-B becomes the Sender. Program-B is then in the initial state (connected) as if the transmission channel just opened, and expects to receive a service ready greeting.

3.2 Fanout and Queued Transactions

An IRIP server must be able to fanout requests targeted at other IRIP servers. An IRIP server may queue information targeted at other IRIP servers. There are several reasons for queing requests. One reason is that firewall issues may prevent one server from contacting another. IRIP provides a SWITCH command described later in this document to help address this situation.

IRIP servers can establish trust relationships between each other. A trusted relationship means:

- one server must authenticate with the other
- authenticated calendars on one server are trusted and treated as authenticated on the other.

The trusted relationship need not be bi-directional. That is, the fact that IRIP server A trusts IRIP server B does not necessarily mean that B trusts A.

A trusted relationship between two IRIP servers means that one server can queue transactions for the other server and deliver them some time later. If IRIP server B trusts A, then A can queue requests for B. If A does not trust B then B cannot accumulate requests for A.

Certain requests may need to be delivered and replied to in real-time. In fact, a requester may wish to cancel the request if the reply cannot be delivered in real-time. In IRIP it is possible to detect whether or not a reply will be made in real-time and cancel the request if necessary.

3.3 Reply Codes

[IRIP] error codes follow the format defined for Status Replies in [ITIP]. All Status Replies as defined in [ITIP] are valid error codes

when returned by an [IRIP] command.

In addition to those defined in [ITIP], [IRIP] defines the following error codes:

REPLY CODE	DESCRIPTOR	MEANING
2.0.1	**START-ICALDATA**	Start ICAL input; end with <CRLF>.<CRLF>
2.0.2	**REPLY-PENDING**	A timeout has occurred. The server is still working on the reply. Use CONTINUE to continue waiting for the reply or ABORT to terminate the command.
2.0.3	**ABORTED**	In response to the client issuing an ABORT command, this reply code indicates that any command currently underway was successsfully aborted.
2.0.4	**TRUSTED-WILL-ATTEMPT**	The specified Calendar is not here but a trust relationship exists between this server and the server on which the Calendar exists. An

		attempt will be made to deliver the request or reply to the Calendar anyway.
2.0.5	**TRUSTED-WILL-QUEUE**	The specified Calendar cannot be contacted directly. The request or reply will be queued and delivered to the target calendar when its IRIP server contacts this server and issues the SWITCH command.
2.0.6	**WILL-ATTEMPT**	The specified Calendar is not here but an attempt will be made to deliver the request or reply to the Calendar anyway.
2.0.7	**QUEUED**	The request or reply has been queued for delivery.
8.0	**GENERAL FAILURE**	A failure has occurred in the Receiver that prevents the operation from succeeding.
8.1	**SERVER TOO BUSY**	Sent when a session cannot be established because the [IRIP] Receiver is too busy.
8.2	**ICAL OBJECT TOO BIG**	Used to signal that an ICAL object has exceeded the server's size limit.
8.3	**DATE TOO LARGE**	A DATETIME value was too large to be represented on this Calendar.

8.4	**DATE TOO SMALL**	A DATETIME value was too far in the past to be represented on this Calendar.
9.0	**INVALID IRIP COMMAND**	An unrecongnized command was received.
10.1	**REFERRAL**	Accompanied by an alternate address. The RECIPIENT specified should be contacted at the given alternate address. The referral address MUST follow the reply code.

10.2	**SERVER SHUT DOWN**	The server is shutting down.
10.3	**SERVER STOPPING FLOOD**	
10.4	**EXCEEDED QUOTAS**	
10.5	**QUEUED TOO LONG**	The ITIP message has been queued too too long. Delivery has been aborted.

4 Implementation Considerations

It is strongly recommended that when an IRIP implementation encounters an error requiring the communication channel between the Sender and Receiver to be dropped that the DISCONNECT command be issued rather than simply breaking the communication channel.

[Editors note: What is the expectation for calstore recipients that don't exist on this server?]

5 Security Considerations

The security of [IRIP] with [SASL] support is highly dependent on the mechanism used to authenticate the client and whether or not the security layer is further negotiated. Without a robust security layer, [IRIP] transactions are subject to eavesdropping and the integrity of [IRIP] transactions may be compromised. Since [IRIP] is designed specifically for real time Internet transactions, it is recommended that implementations use the highest degree of authentication and transmission security possible.

Authentication is fundamental to [IRIP]. It is the basis for granting and denying access. Without a robust security layer [IRIP] will be subject to many possible attacks and the full contents of the server itself may be at risk.

5.1 SASL ANONYMOUS Mechanism

Implementing support for the Anonymous [SASL] significantly increases the vulnerability of the calendar server and its data. Refer to [ANON-SASL] for further information on many threats specific to Anonymous [SASL] access.

5.2 SASL Profile Definition for the protocol

The implementation of [SASL] in [IRIP] requires the server and client

to comply with the following profile extension:

- AUTHENTICATE command.
- Full description of the challenge/response definition.
- Starting octet.
- Authorization identity supplied by the sender must the one used to grant or denied the requested operation.

5.3 Security Threats

5.3.1 Eavesdropping

If [SASL] is used to negotiate a security layer with the server, then traffic is no longer in the clear and eavesdropping will not be restricted.

5.3.2 Connection Flooding

Connections that have not been authenticated within a configurable number of seconds should be disconnected.

5.3.3 Denial of Service Attacks

[Editors note: need explanation and recommendation: ???]

6 Examples

6.1 Unauthenticated Freebusy Request

This examples shows an anonymous request for the freebusy time of irip://cal.example.com/sman. Note that once xyz is authenticated on the irip server either the fully qualified IRIP CALID or the relative CALID can be used to reference a Calendar. That is, "irip://cal.example.com/xyz" and "xyz" refer to the same calendar and can be used interchangeably.

```
R: <listen on TCP port 5228>
S: <establish a TCP connection to cal.example.com port 5228>
R: 2.0
S: AUTHENTICATE ANONYMOUS xyz
R: 2.0
S: RECIPIENT:sman
R: 2.0
S: ICALDATA
R: 2.0.1
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
S: PRODID:-//ACME/DesktopCalendar//EN
S: METHOD:REQUEST
S: VERSION:2.0
```

```
S: BEGIN:VFREEBUSY
S: ORGANIZER:xyz
S: ATTENDEE:sman
S: DTSTAMP:19971113T190000Z
S: DTSTART:19971115T160000Z
S: DTEND:19971116T040000Z
S: UID:www.example.com-873970198738777@host.com


Courtemanche/Mansour/O'Leary      13                  Expires: May 1999
Internet Draft                    IRIP                November 19, 1998


S: END:VFREEBUSY
S: END:VCALENDAR
S: .

<server looks up the freebusy time and builds a reply>

R: Content-Type:text/calendar; method=REPLY; charset=US-ASCII
R: Content-Transfer-Encoding: 7bit
R:
R: BEGIN:VCALENDAR
R: PRODID:-//EXAMPLE/DesktopCalendar//EN
R: METHOD:REPLY
R: VERSION:2.0
R: BEGIN:VFREEBUSY
S: ORGANIZER:irip://cal.example.com/xyz
R: ATTENDEE:irip://cal.example.com/sman
R: DTSTAMP:19971113T190005Z
R: DTSTART:19971115T160000Z
R: DTEND:19971116T040000Z
R: UID:www.example.com-873970198738777@host.com
R: FREEBUSY:19971113T230000Z/PT1H,19971114T210000Z/PT30M
R: END:VFREEBUSY
R: END:VCALENDAR
R: .
R: sman 2.0
R: .
S: DISCONNECT
R: 2.0
R: <disconnect>
S: <disconnect>
```

6.2 Using Switch

This session demonstrates how a poll can be accomplished using the SWITCH command. In this case, the sender (S) becomes the receiver (R) after issuing the switch command.

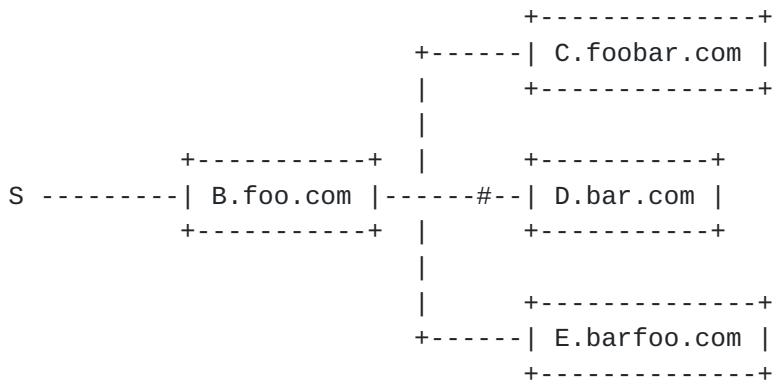
```
R: <listen on TCP port 5228>
```

S: <establish a connection to TCP port 5228>
R: 2.0
S: AUTHENTICATE KERBEROS_V4 93407205
S: <more authentication information>
R: 2.0
S: SWITCH
R: 2.0
<sender now becomes the receiver>
S: 2.0
R: AUTHENTICATE KERBEROS_V4 27367ao986pq8u98u9e8w0-0--0--0werg
S: 2.0
<receiver can now authenticate and send anything pending for the
sender>

6.3 Queued Requests

In the diagram below, sender S has authenticated to the IRIP server B.foo.com. C.foobar.com, D.bar.com, and E.barfoo.com all have IRIP servers. B has a trusted relationship with both C and D. A firewall is in place that prohibits B from initiating a connection to D. However, D

can connect to B.



6.3.1 Meeting Invitation

In this example, S sends an event request to the IRIP server on B for calendars on B, C, D, and E.

R: <listen on TCP port 5228>
S: <establish a TCP connection to b.foo.com port 5228>
R: 2.0
S: AUTHENTICATE KERBEROS_V4 93407205
S: <more authentication information>
R: 2.0
S: RECIPIENT:irip://B.foo.com/bill

```
R: 2.0
S: RECIPIENT:irip://C.foobar.com/cathy
R: 2.0.4
S: RECIPIENT:irip://D.bar.com/david
R: 2.0.5
S: RECIPIENT:irip://E.barfoo.com/eddie
R: 2.0.6
S: ICALDATA: 16
R: 2.0.1
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
S: PRODID:-//ACME/DesktopCalendar//EN
S: METHOD:REQUEST
S: VERSION:2.0
S: BEGIN:VEVENT
S: ORGANIZER:irip://B.foo.com/bill
S: ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:irip://B.foo.com/bill
S: ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:irip://C.foobar.com/cathy
S: ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:irip://D.bar.com/david
S: ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:irip://E.barfoo.com/eddie
S: DTSTAMP:19981011T190000Z
S: DTSTART:19981101T200000Z
S: DTEND:19981101T2100000Z
S: SUMMARY:Conference
S: UID:calsrv.example.com-873970198738777@example.com
S: SEQUENCE:0
S: STATUS:CONFIRMED
S: END:VEVENT
S: END:VCALENDAR
S: .
R: .
R: irip://B.foo.com/bill 2.0
```



```
R: irip://C.foobar.com/cathy 2.0
R: irip://D.bar.com/david 2.0.7
R: irip://E.barfoo.com/eddie 2.0
S: DISCONNECT
R: 2.0
R: <disconnect>
S: <disconnect>
```

The invitation is written to the calendar B.foo.com/bill. IRIP server B.foo.com authenticates to C.foobar.com and sends the event request, which is successfully written to C.foobar.com/cathy. The IRIP server on

B.foo.com cannot contact D.bar.com, but a trust relationship exists between them and the request is queued for delivery. This request will be delivered the next time the IRIP server on D.bar.com connects to the IRIP server on B.foo.com and issues a SWITCH command. The IRIP server on B.foo.com connects to the IRIP server on E.barfoo.com and authenticates as anonymous since it has no trust relationship with E.barfoo.com. If the anonymous authentication is successful, the event request is delivered to E.barfoo.com/eddie.

[Editors note: in the case of the anonymous authentication, B could also provide E with the same credentials as S provided to B]

6.3.2 Busy Time Request

In this example, the sender S sends a Freebusy request to B for calendars on B, C, D, and E. S needs the information immediately and will abort any attempt to queue requests.

```
R: <listen on TCP port 5228>
S: <establish a TCP connection to cal.example.com port 5228>
R: 2.0
S: AUTHENTICATE KERBEROS_V4 93407205
S: <more authentication information>
R: 2.0
S: RECIPIENT:irip://B.foo.com/bill
R: 2.0
S: RECIPIENT:irip://C.foobar.com/cathy
R: 2.0.4
S: RECIPIENT:irip://D.bar.com/david
R: 2.0.5
S: RECIPIENT:irip://E.barfoo.com/eddie
R: 2.0.6

<the sender cannot accept a queued request and response. The
 current operation will be canceled. The operation will be tried
 again with all attendees that have requests queued dropped from the
 RECIPIENT list...>

S: ABORT
R: 2.0.3
S: RECIPIENT:irip://B.foo.com/bill
R: 2.0
S: RECIPIENT:irip://C.foobar.com/cathy
R: 2.0.4
S: RECIPIENT:irip://E.barfoo.com/eddie
R: 2.0.6
S: ICALDATA
R: 2.0.1
```

```
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
S: PRODID:-//ACME/DesktopCalendar//EN
S: METHOD:REQUEST
S: VERSION:2.0
S: BEGIN:VFREEBUSY
S: ORGANIZER:irip://B.foo.com/bill
S: ATTENDEE:irip://B.foo.com/bill
S: ATTENDEE:irip://C.foobar.com/cathy
S: ATTENDEE:irip://D.bar.com/david
S: ATTENDEE:irip://E.barfoo.com/eddie
S: DTSTAMP:19971113T190000Z
S: DTSTART:19971115T160000Z
S: DTEND:19971116T040000Z
S: UID:www.example.com-873970198738777@host.com
S: END:VFREEBUSY
S: END:VCALENDAR
S: .

<server looks up the freebusy time for B.foo.com/bill,
 requests and receives the freebusy time for
 irip://C.foobar.com/cathy and irip://E.barfoo.com/eddie. Then it
 builds a reply>

R: Content-Type:multipart/mixed;boundary="--FEE3790DC7E35189CA67CE2C"
R:
R: This is a multi-part message in MIME format.
R:
R:----FEE3790DC7E35189CA67CE2C
R: Content-Type:text/calendar; method=REPLY; charset=US-ASCII
R: Content-Transfer-Encoding: 7bit
R:
R: BEGIN:VCALENDAR
R: PRODID:-//EXAMPLE/DesktopCalendar//EN
R: METHOD:REPLY
R: VERSION:2.0
R: BEGIN:VFREEBUSY
S: ORGANIZER:irip://B.foo.com/bill
R: ATTENDEE:irip://B.foo.com/bill
R: DTSTAMP:19971113T190005Z
R: DTSTART:19971115T160000Z
R: DTEND:19971116T040000Z
R: UID:www.example.com-873970198738777@host.com
R: FREEBUSY:19971115T200000Z/PT1H,19971116T170000Z/PT30M
R: END:VFREEBUSY
R: END:VCALENDAR
```

```
R:
R:----FEE3790DC7E35189CA67CE2C
R: Content-Type:text/calendar; method=REPLY; charset=US-ASCII
R: Content-Transfer-Encoding: 7bit
R:
R: BEGIN:VCALENDAR
R: PRODID:-//EXAMPLE/DesktopCalendar//EN
R: METHOD:REPLY
R: VERSION:2.0
R: BEGIN:VFREEBUSY
S: ORGANIZER:irip://B.foo.com/bill
```



```
R: ATTENDEE:irip://C.foobar.com/cathy
R: DTSTAMP:19971113T190005Z
R: DTSTART:19971115T160000Z
R: DTEND:19971116T040000Z
R: UID:www.example.com-873970198738777@host.com
R: FREEBUSY:19971115T230000Z/PT1H,19971116T210000Z/PT30M
R: END:VFREEBUSY
R: END:VCALENDAR
R:
R:----FEE3790DC7E35189CA67CE2C
R: Content-Type:text/calendar; method=REPLY; charset=US-ASCII
R: Content-Transfer-Encoding: 7bit
R:
R: BEGIN:VCALENDAR
R: PRODID:-//EXAMPLE/DesktopCalendar//EN
R: METHOD:REPLY
R: VERSION:2.0
R: BEGIN:VFREEBUSY
S: ORGANIZER:irip://B.foo.com/bill
R: ATTENDEE:irip://E.barfoo.com/eddie
R: DTSTAMP:19971113T190005Z
R: DTSTART:19971115T160000Z
R: DTEND:19971116T040000Z
R: UID:www.example.com-873970198738777@host.com
R: FREEBUSY:19971115T230000Z/PT1H,19971116T210000Z/PT30M
R: END:VFREEBUSY
R: END:VCALENDAR
R:
R:----FEE3790DC7E35189CA67CE2C
R: .
R: irip://B.foo.com/bill 2.0
R: irip://C.foobar.com/cathy 2.0
R: irip://E.barfoo.com/eddie 2.0
S: DISCONNECT
```

```
R: 2.0
R: <disconnect>
S: <disconnect>
```

Since each reply is delivered immediately, the reply is sent as a Multipart/Mixed. Transport reply codes for each recipient are returned after the Multipart/Mixed data.

6.4 Resource Scheduling

```
R: <listen on TCP port 5228>
S: <connect to port 5228>
R: 2.0
S: AUTHENTICATE KERBEROS_V4 7SF8S3&#
S: <more authentication information>
R: 2.0
S: RECIPIENT Large_Conference_Room
R: 2.0
S: RECIPIENT Overhead_Projector_1
R: 2.0
S: ICALDATA: 10
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
```



```
S: PRODID:-//Foo Corporation//Inlet MIMEDIR//EN
S: VERSION:2.0
S: METHOD:REQUEST
S: BEGIN:VEVENT
S: ORGANIZER:irip://cal.example.com/xyz
S: ATTENDEE:irip://cal.example.com/Large_Conference_Room
S: ATTENDEE:irip://cal.example.com/Overhead_Projector_1
S: DTSTART:19980706T190000Z
S: DTEND:19980706T203000Z
S: LOCATION:Large Conference Room
S: DESCRIPTION:Big customer meeting in Large Conference room.
S: SUMMARY:Customer meeting
S: PRIORITY:1
S: END:VEVENT
S: END:VCALENDAR
S: .
R: <looks up availability in database>
R: .
R: 2.0 Large_Conference_Room
R: 2.0 Overhead_Projector_1
```

```
S: RECIPIENT Large_Conference_Room
R: 2.0
S: RECIPIENT Overhead_Projector_1
R: 2.0
S: ICALDATA: 10
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
S: PRODID:-//Foo Corporation//Insight MIMEDIR//EN
S: VERSION:2.0
S: METHOD:REQUEST
S: BEGIN:VEVENT
S: ORGANIZER:irip://cal.example.com/xyz
S: ATTENDEE:irip://cal.example.com/Large_Conference_Room
S: ATTENDEE:irip://cal.example.com/Overhead_Projector_1
S: DTSTART:19980708T220000Z
S: DTEND:19980708T230000Z
S: LOCATION:Large Conference Room
S: DESCRIPTION:Another big customer meeting in Large Conference room.
S: SUMMARY:Customer meeting
S: PRIORITY:1
S: END:VEVENT
S: END:VCALENDAR
S: .
R: <looks up availability in database>
R: .
< Large_Conference_Room not available, thus the response code is...>
R: 4.0
S: RECIPIENT Large_Conference_Room
R: 2.0
S: RECIPIENT Overhead_Projector_1
R: 2.0
S: ICALDATA: 10
S: Content-Type:text/calendar; method=REQUEST; charset=US-ASCII
S: Content-Transfer-Encoding: 7bit
S:
S: BEGIN:VCALENDAR
S: PRODID:-//Foo Corporation//Insight MIMEDIR//EN
```



```
S: VERSION:2.0
S: METHOD:REQUEST
S: BEGIN:VEVENT
S: ORGANIZER:irip://cal.example.com/xyz
S: ATTENDEE:irip://cal.example.com/Large_Conference_Room
S: ATTENDEE:irip://cal.example.com/Overhead_Projector_1
```

```
S: DTSTART:19980708T160000Z
S: DTEND:19980708T170000Z
S: LOCATION:Large Conference Room
S: DESCRIPTION:Another big customer meeting in Large Conference room.
S: SUMMARY:Customer meeting
S: PRIORITY:1
S: END:VEVENT
S: END:VCALENDAR
S: .
R: <looks up availability in database>
<10 seconds pass>
R: .
<Database too busy, thus the response code is...>
R: 2.0.2
S: ABORT
R: 2.0
S: DISCONNECT
R: 2.0
R: <drops TCP connection>
```

7 Acknowledgments

The following have participated in the drafting and discussion of this memo:

Bruce Kahn, Doug Royer, Mugino Saeki

8 Bibliography

[ICAL] "Internet Calendaring and Scheduling Core Object Specification - iCalendar", Internet-Draft, July 1997, ftp://ftp.ietf.org/internet-drafts/draft-ietf-calsch-ical-10.txt.

[ICMS] "Internet Calendaring Model Specification", Internet-Draft, July 1997, ftp://ftp.ietf.org/internet-drafts/draft-ietf-calsch-mod-00.txt.

[ITIP] "iCalendar Transport-Independent Interoperability Protocol (iTIP) : Scheduling Events, Busy Time, To-dos and Journal Entries ", Internet-Draft, October 1997, http://www.imc.org/draft-ietf-calsch-itip-06.txt.

[IMIP] "iCalendar Message-based Interoperability Protocol (iMIP), Internet-Draft, October 1997, http://www.imc.org/draft-ietf-calsch-imip-05.txt.

[ID-UTF8] "UTF-8, a transformation format of Unicode and ISO 10646", Internet-Draft, July,1996, ftp://ftp.ietf.org/internet-drafts/draft-yergeau-utf8-01.txt.

[RFC-822] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

[RFC-1847]. J. Galvin, S. Murphy, S. Crocker & N. Freed, "Security

Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

[RFC-2112] Levinson, E., "The MIME Multipart/Related Content-type," RFC 2112, March 1997.

[RFC-2015] M. Elkins, "MIME Security with Pretty Good Privacy (PGP)," RFC 2015, October 1996.

[RFC-2045] Freed, N., Borenstein, N., " Multipurpose Internet Mail Extensions (MIME) - Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC-2046] Freed, N., Borenstein, N., " Multipurpose Internet Mail Extensions (MIME) - Part Two: Media Types", RFC 2046, November 1996.

[RFC-2047] Moore, K., "Multipurpose Internet Mail Extensions (MIME) - Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[RFC-2048] Freed, N., J. Klensin, J. Postel, "Multipurpose Internet Mail Extensions (MIME) - Part Four: Registration Procedures", RFC 2048, January 1997.

[RFC-2222] J. Meyers, Simple Authentication and Security Layer (SASL)", RFC 2222, October 1997.

9 Open Issues

Registration of the [SASL] profile for [IRIP] with the IANA.
Port Number registration

10 Author's Address

The following address information is provided in a vCard v2.1, Electronic Business Card, format.

BEGIN:VCARD
FN:Andre Courtemanche
ORG:CS&T
ADR;WORK;POSTAL;PARCEL:;;3333 Graham Boulevard;Montreal;QC;H3R 3L5;Canada

TEL;WORK;MSG:+1-514-733-8500
TEL;WORK;FAX:+1-514-733-8788

```
EMAIL;INTERNET:andre@cst.ca
END:VCARD

BEGIN:VCARD
VERSION:2.1
FN:Steve Mansour
ORG:Netscape Communications Corporation
ADR;WORK;POSTAL;PARCEL:;;501 East Middlefield Road;Mountain
  View;CA;94043;USA
TEL;WORK;MSG:+1-650-937-2378
TEL;WORK;FAX:+1-650-937-2103
EMAIL;INTERNET:sman@netscape.com
END:VCARD
```



```
BEGIN:VCARD
FN:Pete O'Leary
ORG:Amplitude
ADR;WORK;POSTAL;PARCEL:;;
TEL;WORK;MSG:+1-415-659-3511
TEL;WORK;FAX:+1-415-659-0006
EMAIL;INTERNET:pete@amplitude.com
END:VCARD
```

The iCalendar object is a result of the work of the Internet Engineering Task Force Calendaring and scheduling Working Group. The chairman of that working group is:

```
BEGIN:VCARD
FN:Anik Ganguly
ORG:Open Text, Inc.
ADR;WORK;POSTAL;PARCEL:;;38777 West Six Mile Road Suite 101;
    Livonia;MI;48152;USA
TEL;WORK;MSG:+1-734-542-5955
EMAIL;INTERNET:ganguly@acm.org
END:VCARD
```

The co-chairman of that working group is:

```
BEGIN:VCARD
VERSION:2.1
FN:Robert Moskowitz
EMAIL;INTERNET: rgm-ietf@htt-consult.com
END:VCARD
```

11 Full Copyright Statement